16.482 / 16.561: Computer Architecture and Design

Spring 2015

Midterm Exam
March 5, 2015

Name: _____ **ID #:** _____

For this exam, you may use a calculator and two 8.5" x 11" double-sided page of notes. All other electronic devices (e.g., cellular phones, laptops) are prohibited. If you have a cellular phone, please turn it off prior to the start of the exam to avoid distracting other students.

The exam contains 6 questions for a total of 100 points. Please answer the questions in the spaces provided. If you need additional space, use the back of the page on which the question is written and clearly indicate that you have done so.

You will be provided with two pages (1 double-sided sheet) that contain a list of the MIPS instructions we have covered. You do not have to submit this sheet when you turn in your exam.

You will have three hours to complete this exam.

Q1: Evaluating instructions	/ 8
Q2: Binary multiplication	/ 16
Q3: IEEE floating-point format	/ 20
Q4: Datapaths and pipelining	/ 16
Q5: Dynamic branch prediction	/ 21
Q6: Dynamic scheduling	/ 19
TOTAL SCORE	/ 100

1. (8 points) ***Evaluating instructions***

Assume the following initial state prior to executing the instructions below. Note that the result of each instruction may depend on prior instructions.

- `$t1 = 0x0000180C, $t2 = 0x0000409, $s0 = 0x00121000`
- Contents of memory (all values are in hexadecimal)

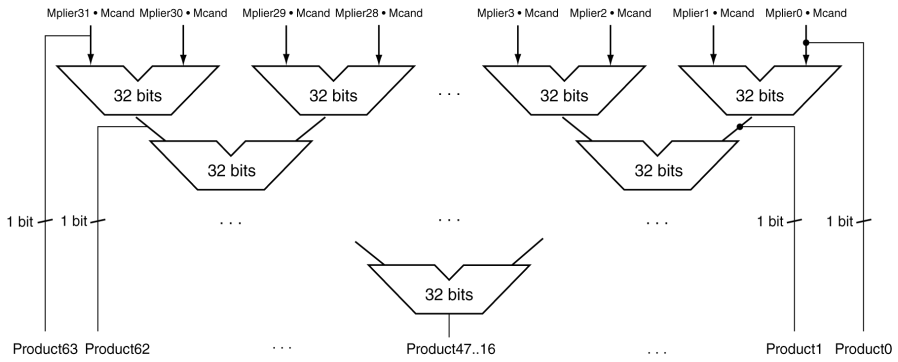
Address	Lo			Hi
0x00121200	11	CB	42	98
0x00121204	31	42	93	AA

For each instruction below, list the changed register or memory location(s) and its new value. Note that constant values are in decimal unless specified as hexadecimal by a leading 0x.

```
   sub  $t3, $t1, $t2

   andi $t3, $t3, 0xF00F

   sb   $t3, 0x203($s0)

   lhu  $t4, 0x206($s0)

   slti $s3, $t1, 0x2000

   beq  $s3, $zero, L

   sll  $t3, $t3, 8

L: addi $t1, $t1, 10
```

2. (16 points) ***Binary multiplication***

a. (6 points) The tree multiplier discussed in class is shown below:



Determine the time required for a tree multiplier to multiply two 64-bit numbers if each step of the operation takes 4 ns. (The figure above describes a 32-bit multiplier.) Note that some of the hardware may be able to operate in parallel.

2 (continued)

b. (10 points) You are given A = 5 and B = 7. Assume each operand uses four bits. Show how the binary multiplication of A * B would proceed using the iterating multiplier discussed in class.

3. (20 points) ***IEEE floating-point format***

Add the two IEEE single-precision floating-point values 0x42960000 and 0x41ea0000. For full credit, you must show all work, including:

- Convert the two values into binary
- Perform the addition in binary, not decimal
- Re-encode the result in IEEE single-precision format

4. (16 points) ***Datapaths and pipelining***

a. (4 points) Explain the differences between a single-cycle datapath and a pipelined datapath.

b. (4 points) Explain how forwarding helps remove data hazards.

4 (continued)

c. (8 points) Consider the following code sequence:

```
lw   $t0, 0($s1)
lw   $t1, 4($s1)
add  $t3, $t0, $t1
xor  $t4, $t0, $t1
sub  $t5, $t3, $t4
beq  $t5, $zero, L1
addi $s1, $s1, 8
sw   $t3, 0($s1)
```

Determine the time required to execute this sequence on a processor with a basic 5-stage pipeline without forwarding. Assume the branch is not taken. Express your answer in cycles, not ns.

5. (21 points) ***Dynamic branch prediction***

a. (14 points) Say you are executing a program that contains two branches, as shown below. You are given the addresses of each branch in both decimal and hexadecimal. Note that these branches are inside a loop, but neither one controls the number of loop iterations.

Address		
Decimal	Hex	
56	0x38	`BEQ $t0, $s0, label1`
		…
68	0x44	`BNE $t5, $t6, label2`

Your processor contains a thirty-two entry, 2-bit branch history table. Initially, entries 0-15 (the first sixteen lines of the table) all have the state 10, and entries 16-31 (the last sixteen lines of the table) all have the state 01.

Complete the table below to show which BHT entry is used to predict each branch, what predictions are made based on that entry, and how the state of each BHT entry changes throughout the program. You are given the actual outcome for each branch.

Loop Iteration	Branch	BHT Entry #	BHT Entry State	Pred.	Actual Outcome	New BHT Entry State
1	BEQ				T	
1	BNE				NT	
2	BEQ				NT	
2	BNE				T	
3	BEQ				T	
3	BNE				T	
4	BEQ				NT	
4	BNE				NT	

5 (continued)

b. (4 points) Determine the number of address bits required to choose the appropriate row in a (4,2) correlating branch predictor with 4096 entries. (Hint: $1024 = 2^{10}$) Show all work to justify your answer.

c. (3 points) Describe a case in which a branch target buffer (BTB) would not hold the target address of the branch being predicted.

6. (19 points) ***Dynamic scheduling***

a. (3 points) Explain how a dynamically scheduled processor without speculation determines if an instruction is allowed to write its result to the register file once it completes.

b. (3 points) Under what conditions would two instructions in a dynamically scheduled processor be allowed to start executing in the exact same cycle without causing any type of hazard?

c. (3 points) In a dynamically scheduled processor with speculation, why are registers renamed based on their reorder buffer entries, not the reservation stations to which the instructions are issued?

d. (10 points) Complete the pipeline diagram below to show how the given code is executed on a dynamically scheduled processor without speculation. Assume the following latencies, which refer to the number of execution cycles unless otherwise noted, and assume that instructions listed separately use different functional units:

- 2 cycles (1 EX, 1 MEM) for L.D and S.D
- 3 cycles for ADD.D and SUB.D
- 8 cycles for DIV.D
- 1 cycle for all other instructions

Also, assume that the processor only contains one common data bus. Note that your solution may not use all 20 cycles shown below, but it should not use more than 20 cycles.

Inst.	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
`L.D F0,0(R1)`																				
`L.D F2,8(R1)`																				
`ADD.D F4,F0,F2`																				
`DADDUI R1,R1,24`																				
`SLT R2,R1,R4`																				
`L.D F6,-8(R1)`																				
`DIV.D F8,F4,F6`																				
`S.D F8,0(R1)`																				
`BNE R2,R0,Loop`																				